Astronomical Data Analysis Software and Systems VI
ASP Conference Series, Vol. 125, 1997
Gareth Hunt and H. E. Payne, eds.

Data Format for the NOAO Mosaic

Francisco Valdes

IRAF Group, NOAO,[1] *PO Box 26732, Tucson, AZ 85726*

Abstract. NOAO is building a large format (8K×8K pixels) camera using a mosaic of eight 2048×4096 CCDs with eight amplifier readouts initially, and eventually up to 16. This paper outlines the data format and keyword definition methodology for data from this instrument. The software system is described in other papers (see Tody 1997; Valdes 1997).

1. The Data File Structure

NOAO CCD Mosaic data is stored as a single FITS file for each observation. The FITS file contains a *primary header* with no associated data and a number of *extensions*. The primary header describes the file contents, and contains *global* keyword information applicable to all extensions. The extensions include the image data from each amplifier, pixel masks, uncertainty arrays, exposure maps, auxiliary tables, etc. The image data extensions are always present, and other information is added during the reductions.

Figure 1 shows the data file structure, and illustrates how the *inheritance* convention (Zarate & Greenfield 1995) defines the header for each image extension as the combination of the global and individual header keywords.

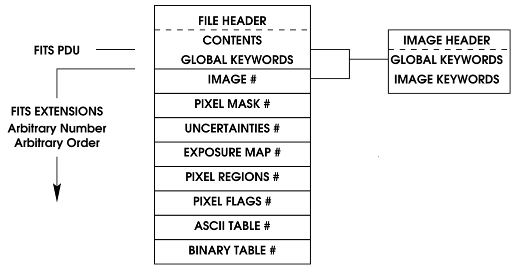

Figure 1. Mosaic Data File Structure.

The Image Data The basic image data consist of separate FITS Image Extensions for each amplifier. Each has an *extension name* used by the IRAF software to refer to the image through the *IRAF FITS Image Kernel* (Zarate

[1]National Optical Astronomy Observatories, operated by the Association of Universities for Research in Astronomy, Inc. (AURA) under cooperative agreement with the National Science Foundation.

& Greenfield 1995) (e.g., "`obs123[im3]`" refers to the image data for the third amplifier). The pixel masks, uncertainty arrays, and other array type extensions are also accessed through the FITS kernel by extension name.

Pixel Masks *Pixel masks* assign an non-negative integer value to each pixel in an image. The meaning of the mask value depends on the purpose of the mask—there may be more than one assigned to an image—and the application that will use it. Because it is often the case that most pixels have the same mask value IRAF provides a special representation called a *pixel list.* This representation is very compact. Some types of real data, such as the uncertainty array (described in the next section), may also consist of regions of constant value or be usefully mapped to integers using something like the BSCALE and BZERO method in FITS. The pixel list is stored as a FITS extension in a form still to be defined but most likely based on a binary table. The FITS kernel will be able to convert this type of extension to a standard IRAF pixel mask so applications may use these without knowledge of the FITS representation.

The types of integer pixel masks that might be used are identification of good and bad pixels with a set of code values for the type of bad pixel (CCD defect, saturated, etc.), the number of input pixels contributing to a pixel in a combined image, data quality flags, and marking regions for various purposes. The types of real pixel masks are uncertainty values and exposure maps showing the accumulated exposure time contributing to a pixel.

Uncertainties An important aspect of the image data is the uncertainties. Many of the concepts are reasonably well understood such as the characterization of the uncertainties in the raw CCD data in terms of a readout noise and Poisson statistics and how uncertainties are propagated when combining pixels with independent errors. Others are less well understood such as what happens with resampling. The biggest dilemma has been how to maintain the uncertainty information without doubling the data volume by using an associated data array of uncertainty values of the same size as the image data. In terms of the data structure we need something that will be compact yet offer the flexibility to characterize the uncertainties of each pixel.

The model we propose for CCD uncertainties is

$$\sigma^2(i,j) = A + (B + I(i,j))) * f(U(i,j)) \tag{1}$$

where $I(i,j)$ is the data, A and B are constants, $U(i,j)$ is an array of values, and f is a mapping function. In order to provide a compact description $U(i,j)$ is represented as a pixel list of integers which, hopefully, have large regions of constant value. The use of integers means that the variances will be quantized at some precision. The mapping function f can be defined to adjust the resolution at different levels. Note that there is already a mapping relative to the pixel sigmas because of the definition in terms of the variance.

This model allows easy propagation of errors in the common cases. The A value is a constant noise term. Typically this would be the CCD readout noise. When adding or subtracting two images, the corresponding A terms add. The B term is used when adding or subtracting constant values from images. For raw CCD data this value is zero. The usefulness and compactness of this model, that is how well the idea of largely constant areas in the U array will work in practice, still needs to be investigated. Preliminary experiments show promise that this approach will work effectively.

2. The Observation Information

The basic observational data consist of CCD pixel values and descriptive or documentary information associated with the observation. Pixel values are recorded as separate FITS image rasters for each amplifier. Observation information is recorded in FITS headers, both the primary global header and the extension image headers. This section outlines the method used to identify the interesting observation information and translate it into FITS keywords. Complete details are given on the NOAO Keyword Definitions[2] page.

There are three steps to defining the content of the observation information to be stored in the data file: (*i*) define a *logical model* of the observation information that includes **all** relevant or interesting items, (*ii*) translate the information into well defined and documented FITS keywords, and (*iii*) to collect the information and place it into the observation data file using the keyword definitions. The information actually recorded will be a subset of all the logical observation information identified in the first step.

The Logical Model The logical model of an observation attempts to identify **all** the information about an observation in a systematic manner. This model and framework is general for all ground-based optical and near-IR data and may be adopted and extended by other observatories.

The logical model analyzes an observation into a hierarchy of *classes* modeled after the logical components of the astronomical sources, the instrumentation, the data format, and archiving. A class consists of information elements which are either individual pieces of information or instances of another class. An element may also be an array of one or more instances such as, for example, information about multiple objects in the field of view.

In this short paper we can only present a brief example to give a flavor of this methodology. The root class is OBSERVE. It consists mostly of other classes such as OBJECT[n], TELESCOPE, INSTRUMENT, and DETECTOR. Note that the OBJECT class can have multiple instances. Many classes, such as the those previously mentioned, include general subclasses such as COORDINATE. A COORDINATE class has elements such as right ascension, declination, and system. This not only allows multiple coordinates for things like objects and telescope pointings but also different coordinate system types and equinoxes.

The COORDINATE class example shows one of the powers of this methodology. In the class we define an element such as equinox. Then we can be assured that any coordinate included as an element of some other class will have the equinox explicitly included and not forget that a coordinate must include this to be complete.

An information element has a hierarchical identifier. Examples of these are "`Object[n].Coordinate.ra`" and "`Detector.Ccd[n].Amp[n].Exp.darktime`." In words the latter example says there is a piece of exposure information that applies to a particular amplifier, in a particular CCD, in a detector which gives the effective dark current time. The capitalized words are instances of classes and shows that darktime is a node element of the Exp class which is a subclass element of the Amp class and so on.

---

[2]http://iraf.noao.edu/docs/keywords/

In our draft logical model there are 45 classes. The classes have anywhere from two to ten elements many of which are instances of another class. This is a manageable description even though if we expand out all possible elements as identifiers we get a very large number of elements. This leads us to be confident that we have identified all the pieces of information which we would not have if we started by trying to define all the leaf elements directly.

Clearly it is not possible to define all the information for every instrument and type of observation. However, the logical class model can be extended in a systematic way. This can be done by adding additional elements to a class or adding new classes. Instrument or system specific classes, such as for a particular instrument or array controller, may be added to define parameters which do not fit the general observation model. After the logical model is extended then the mapping to FITS keywords can be made.

FITS Keywords This section outlines the mapping of the logical observation identifiers to FITS keywords. The logical model is very general and could be used by many observatories. The mappings to FITS keywords could be more observatory specific though the same mappings could also be used by different observatories.

Every piece of information identified by the logical model has both a logical name and a FITS keyword. The mapping is given in a *keyword dictionary*. The keyword dictionary not only defines the keywords but acts as additional and more detail documentation about the meanings and assumptions for the information recorded in the image headers.

The keyword dictionary gives the logical identifier, the FITS keyword, a substitute keyword, the units, comment string, definition, etc. It is the substitute keyword that allows mapping the very large number of logical elements to a far few number of keywords. Note that there is no requirement that all the defined keywords appear in the FITS header. There are several reasons why items will not appear. Some items do not make sense for particular instruments, some items may not be available to the data acquisition system, and items with identical values may be mapped to a common keyword.

While the logical header identifies each possible item separately, many items will have the same value. These can be mapped to a single FITS keyword through the substitute keyword entry of the dictionary. An example of this is the coordinate system identification which may be the same for all coordinates; i.e., all coordinates are given in FK5 with equinox J2000. Items may also map to the same keyword because there is no precise value but a related value is approximately correct. An example of this is if the location of the center of the detector on the sky is not known then the telescope position my be substituted.

References

Tody, D. 1997, this volume, 451

Valdes, F. 1997, this volume, 455

Zarate, N., & Greenfield, P. 1995, in ASP Conf. Ser., Vol. 101, Astronomical Data Analysis Software and Systems V, ed. G. H. Jacoby & J. Barnes (San Francisco: ASP), 331